MATLAB TUTORIAL 2 Part 1

I saw many examples (mostly involving loops) to simulate athletes / drug takers. Some codes looked more efficient (e.g. shorter, easier to follow) than others.

Most people got this to work (i.e. in line with the theoretical result, so mean of the simulated data approximately 0.31, as given by Bayes' Theorem). If you didn't get the right result, go back and read all the instructions carefully before writing your code.

MATLAB TUTORIAL 2 Part 1

Theoretical probability of taking drugs given tested positive on one test

Define events:

Let D denote took drugs.

Let T denote positive test.

Given $P(T|D) = 0.8$, $P(T|\overline{D}) = 0.2$, $P(D) = 0.1$.

TTP and Bayes' Theorem:

$$P(D|T) = \frac{P(T|D)P(D)}{P(T)} = \frac{0.8 * 0.1}{0.8 * 0.1 + 0.2 * 0.9} = 0.31$$

MATLAB TUTORIAL 2 Part 1

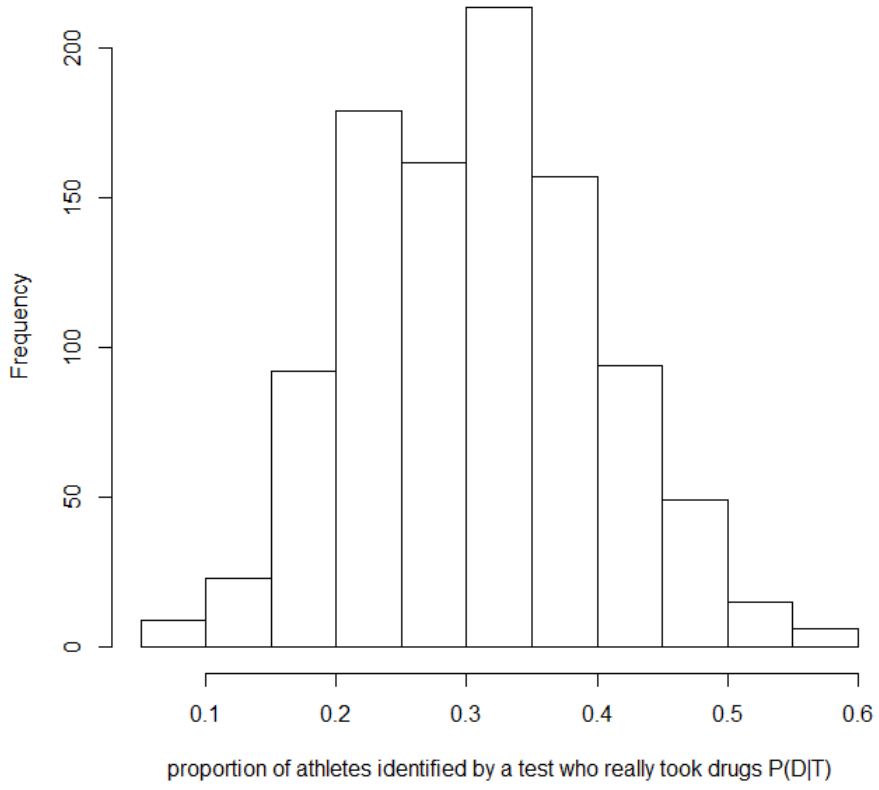

Figure: Simulated conditional probabilities of drugs given positive test in 100 athletes (1000 simulations)

MATLAB TUTORIAL 2 Part 1

Theoretical probability of taking drugs given tested positive on two tests

Define events:

Let D denote took drugs.

Let T_1, T_2 denote positive test 1 and test 2.

Given $P(T_1|D) = P(T_2|D) = 0.8$, $P(T_1|\overline{D}) = P(T_2|\overline{D}) = 0.2$, $P(D) = 0.1$.

And by conditional independence of the tests given drug status,

$$P(T_1 \cap T_2|D) = P(T_1|D)P(T_2|D)$$

TTP and Bayes' Theorem:

$$P(D|T_1 \cap T_2) = \frac{P(T_1 \cap T_2|D)P(D)}{P(T_1 \cap T_2)}$$

$$= \frac{P(T_1|D)P(T_2|D)P(D)}{P(T_1 \cap T_2)}$$

$$\frac{0.8 * 0.8 * 0.1}{0.8 * 0.8 * 0.1 + 0.2 * 0.2 * 0.9} = 0.64$$

MATLAB TUTORIAL 2 Part 2

I suggest you need not do this exercise for all the distributions in detail, but try the exercise for one discrete and one continuous distribution, and be aware of which other distributions MATLAB provides.

For discrete distributions, plot the theoretical distributions as either very narrow bars (set width to say 0.05) or plot as individual points, rather than joining the points up with a line.

You could use the `stairs` function to plot c.d.f. for discrete random variables.

I have used Poisson and Normal distributions here.

MATLAB TUTORIAL 2 Part 2

Example for Poisson Distribution

`lambda = 4`

`x = 0:20;` create a vector of values to evaluate pmf and cdf

`p = poisspdf(x, lambda);` evaluate pmf at x values

`F = poisscdf(x, lambda);` evaluate cdf at x values

MATLAB TUTORIAL 2 Part 2

Plotting

```
figure(1); clf;

subplot(1, 2, 1);

bar(x, p, 0.05); axis square; xlabel('x');
ylabel('p(x)');

subplot(1, 2, 2);

stairs(x, F); axis square; xlabel('x');
ylabel('F(x)');
```

MATLAB TUTORIAL 2 Part 2

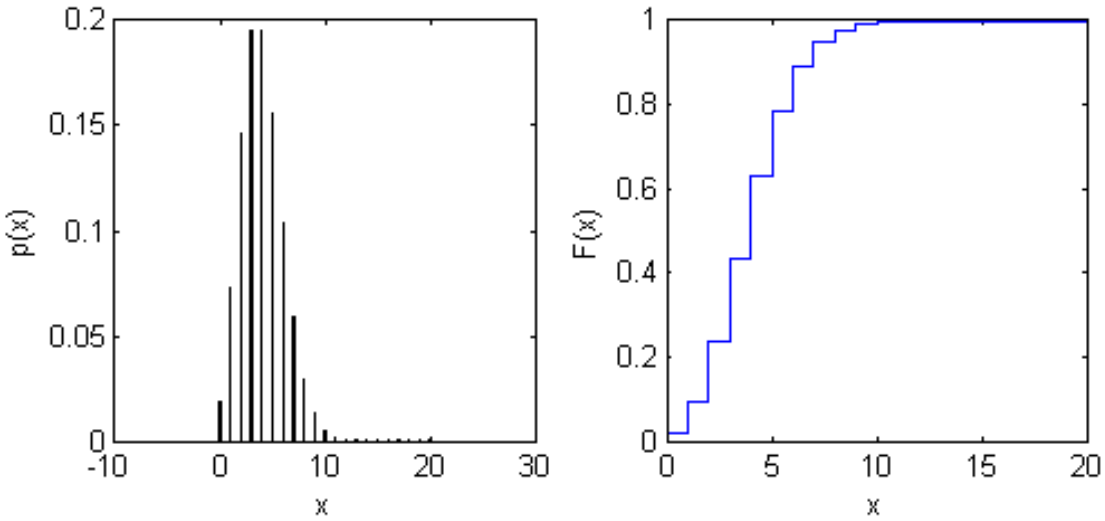

Figure: pmf and cdf for Poisson(4) distribution

MATLAB TUTORIAL 2 Part 2

generate random numbers from Poisson(4) distribution: theoretical mean and variance are both 4.

```
n = 100;

rn = poissrnd(lambda, [1 n]);

samplemean  = mean(rn);

samplevariance = var(rn);
```

My simulation here has mean 3.80 and variance 3.73.

Try for larger values of n - what happens?

MATLAB TUTORIAL 2 Part 2

Then plot empirical probabilities and overlay theoretical probabilities.

First calculate observed probabilities:

```
nx = hist(rn, x); nx = nx./(sum(nx));
```

Then plot

```
figure(1); clf; hold on;
```

```
bar(x, p, 0.05);
```
Theoretical

```
plot(x, nx, 'r*');
```
Observed

repeat for different values of `n` (loop)

MATLAB TUTORIAL 2 Part 2

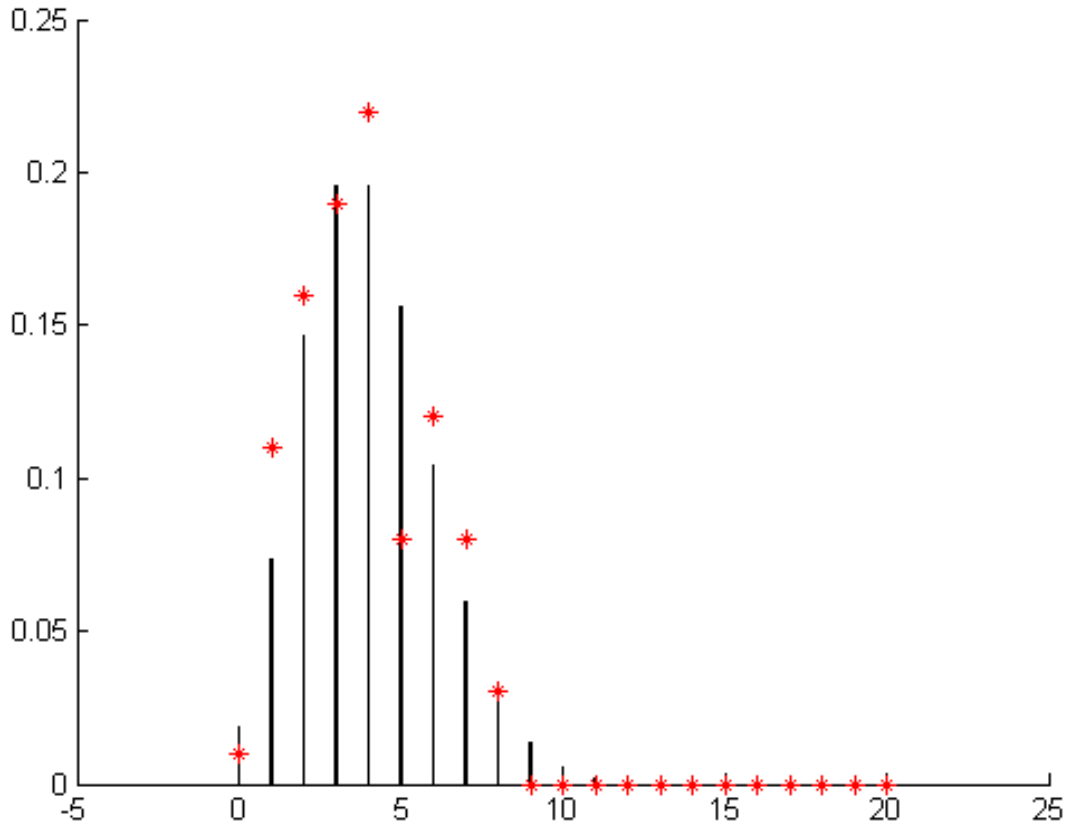

Figure: n = 100 values from Poisson(4) plotted as red stars, compared with theoretical distribution

MATLAB TUTORIAL 2 Part 2

Is it (visually) easier to compare cdfs?

First calculate observed cumulative probabilities:

```
ecdf = cumsum(nx);
```

Then plot

```
figure(1); clf;  hold on;
```

`stairs(x, F);` Theoretical

`plot(x, ecdf, 'r*');` Observed

MATLAB TUTORIAL 2 Part 2

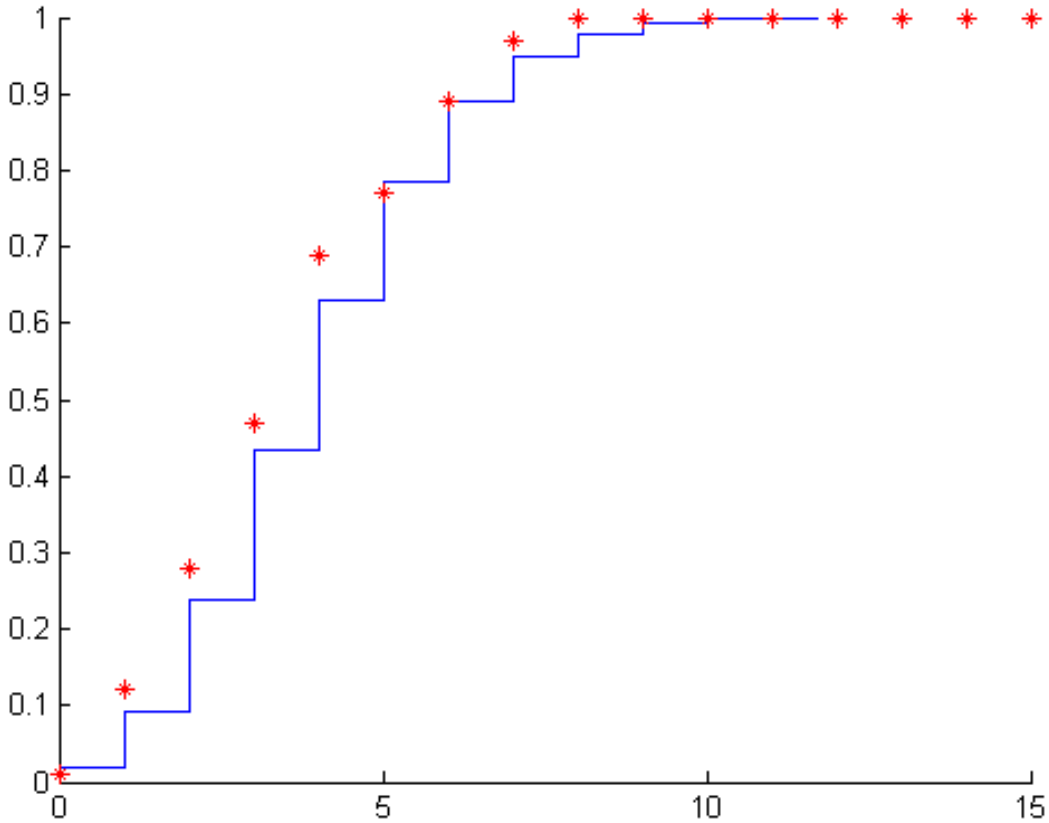

Figure: n = 100 values from Poisson(4) plotted as red stars, compared with theoretical distribution

MATLAB TUTORIAL 2 Part 2

Example for Normal Distribution

```
mu = 0; sigma2 = 1; nx = 100;
```

`x = linspace(-10, 10, nx);` create a vector of values to evaluate pdf and cdf

`f = normpdf(x, mu, sqrt(sigma2));` evaluate pdf at x values

`F = normcdf(x, mu, sqrt(sigma2));` evaluate cdf at x values

MATLAB TUTORIAL 2 Part 2

Plotting

```
figure(1); clf;

subplot(1, 2, 1);

plot(x, f, 'k-'); axis square; xlabel('x');
ylabel('f(x)');

subplot(1, 2, 2);

plot(x, F, 'k-'); axis square; xlabel('x');
ylabel('F(x)');
```

MATLAB TUTORIAL 2 Part 2

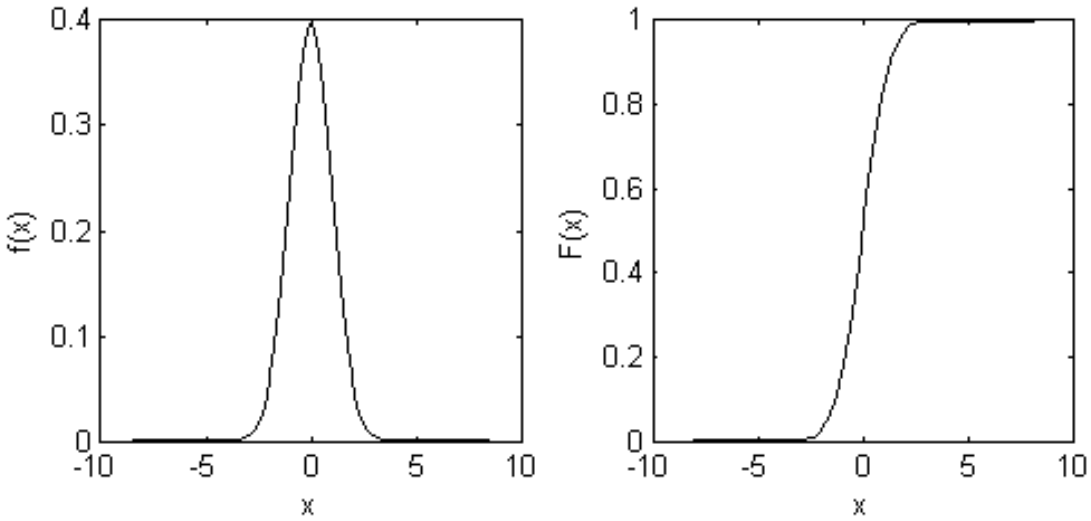

Figure: pdf and cdf for Normal(0,1)

MATLAB TUTORIAL 2 Part 2

generate random numbers from Normal distribution

```
n = 100;

rn = normrnd(mu, sqrt(sigma2), [1 n]);

samplemean = mean(rn);

samplevariance = var(rn);
```

My simulation here has mean -0.0039 and variance 0.8797.

MATLAB TUTORIAL 2 Part 2

Then plot histograms (and overlay densities) - but better to use `histogram` command than `hist` and `bar`. (I do not have the most recent version of MATLAB hence out-of-date code).

```
nx = hist(rn, x);
dx = x(2)-x(1);
nx = nx./(sum(nx)*dx);

hold on;

bar(x, nx, 1, 'FaceColor', 'w');

plot(x, f, 'r-', 'LineWidth', 2);
```

repeat for different values of `n` (loop)

MATLAB TUTORIAL 2 Part 2

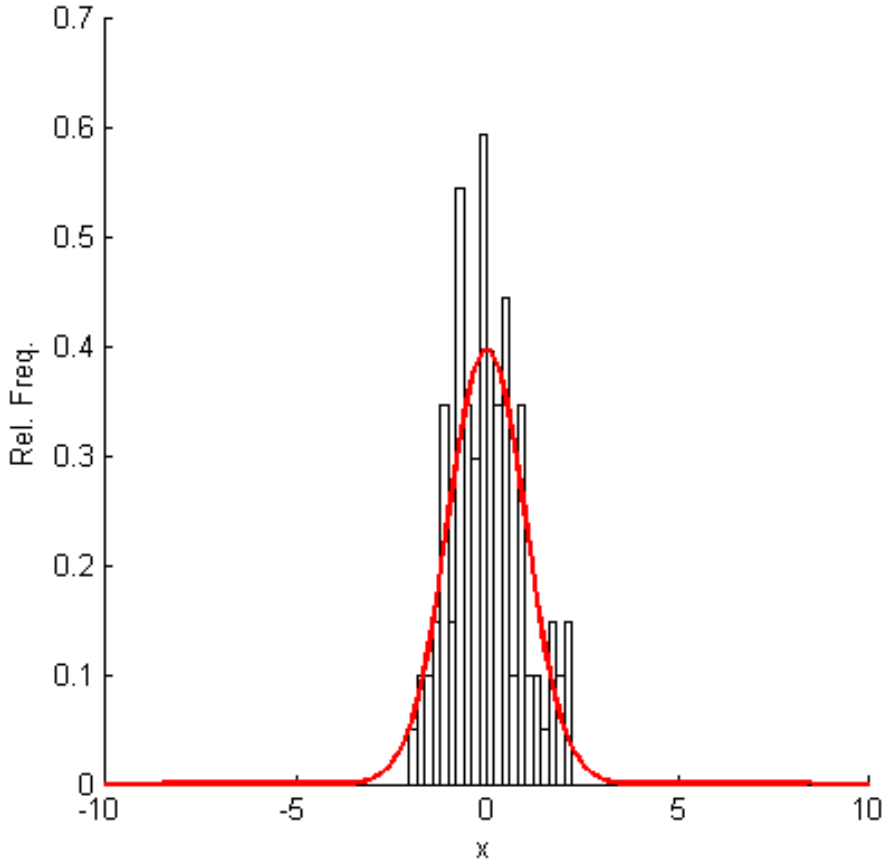

Figure: n = 100 values from Normal(0,1)